# Threat Management Dashboard for a Blockchain Collaborative Defense

Killer, Christian ; Rodrigues, Bruno ; Stiller, Burkhard

# Threat Management Dashboard for a Blockchain Collaborative Defense

Christian Killer, Bruno Rodrigues, Burkhard Stiller
Communication Systems Group CSG, Department of Informatics IfI, University of Zurich UZH
Binzmühlestrasse 14, CH-8050 Zürich
E-mail: [killer,rodrigues,stiller]@ifi.uzh.ch

***Abstract*—The increasing number of Distributed Denial-of-Service (DDoS) attacks leads to a growing research and development interest in DDoS defense systems. As a response to the increasingly distributed nature of DDoS attacks, many organizations have demonstrated willingness to exchange information concerning threats, incidents, and mitigation strategies. Blockchain is, in this regard, a viable threat sharing platform, where different organizations can interact in a verifiable manner. This paper tackles the security visualization and management issue in a blockchain-based collaborative defense platform, defining an interactive dashboard displaying on-going threat mitigation status and enabling security analysts to react on threats on an individual or group level. The dashboard was implemented and evaluated on real hardware emulating the exchange of threats between three isolated Autonomous Systems (AS).**

## I. Introduction

As internet access becomes progressively democratic, including not only many different people using diverse types of stationary and portable devices, there is a concern about device security and the awareness of personal security. In addition to the growing number of connected devices, the increase of available bandwidth allows these devices to exchange a larger volume of data in support of current services and experiences, such as online gaming and video streaming [1]. However, negative side-effects of such a hyper-connected world can be translated into the number of threats and attacks, which are periodically reported by security organizations across the world. Recent statistics on security reports show, for example, not only a steady increase in the number of Distributed Denial-of-Service (DDoS) attacks, but also the number of long-duration attacks (*e.g.*, the most extended attack was longer than 12 days [2]).

Centralized defenses can become a bottleneck due to the need to analyze all traffic measurements at a single location. Thus, the distributed nature of DDoS attacks suggests that a distributed and coordinated defense is the best alternative for a successful defense [3]. The advantages of cooperative defenses over traditional and centralized defenses have been widely recognized in the literature [4], [5]. For instance, they allow for comibing detection/mitigation capabilities of different domains, reduce the detection/mitigation overhead at a single point, and block malicious traffic near its source. However, there is still no widespread deployment of such cooperative defense systems.

In this regard, a permissioned Blockchain (BC) is a trustworthy, decentralized, and publicly available data storage, effectively supporting all members of the cooperative DDoS defence alliance. BC capabilities can be leveraged to build a platform for signaling attacks, serving as an immutable platform for the exchange of mitigation services defined in Smart Contracts (SC) of different peers, but also to provide incentives stimulating the cooperative behavior among service providers [6], [7]. Thus, if an attack is highly sophisticated and there are no countermeasures available, it is possible to request for cooperative mitigation for any domain participating in the alliance.

The major challenge, regardless of the underlying technology, is how to visualize information in a clear and objective manner considering the particularities of a collaborative defense. Thus, it is also required to consider that specialists must analyze not only internal threats, but also external mitigation requests. As the first step to threat mitigation is realizing its existence, it is critical for analysts to use a proper tool to structure and categorize data such that visualization "makes sense" [8]. A collaborative defense involves multi-disciplinary concepts, and the decision-making process usually requires a low response time from the user, but selecting an appropriate type of graphical representation and flow of interaction is not a straightforward task [9].

This paper extends the work proposed in [10], presenting a threat management dashboard for the operation of a collaborative defense. The design considers the basic principles of Security Visualization applied to a collaborative defense, in which the primary stakeholder is a cybersecurity analyst, who can be either in the position of a Mitigator ($M$) or a Target ($T$). The main functional actions within the dashboard are defined according to the role of the organization in the cooperative defense, which is defined as follows: (i) $T$ can request a mitigation or ignore an alarm, or (ii) as a $M$, analyzes incoming requests being accepted or declined. Furthermore, an analyst needs to be able to check service status and start or stop them in a way that is straightforward to manage, for example, from a web browser [11].

This paper is structured as follows. Section II presents related work on security visualization. Section III highlights the requirements that guided the dashboard development. Finally, the use-case evaluation is presented in Section IV, followed by a summary in Section V.

## II. Related Work

A wide range of visualization techniques have been applied in support of network security analytics. Visualization aims at converting collected data into useful information supporting the activities of a cybersecurity analyst [12]. However, techniques have been adapted from traditional network performance and health approaches to security, which are typically not adequately tested concerning their utility or usability providing a Situation Awareness (SA) to security analysts about vulnerabilities and threats [13].

There are different challenges to reach SA: (a) being critical to identify and (b) to model the relevant activities of interest. According to [11], SA can be achieved by recognizing the (i) current situation and acknowledging that an attack occurs, measuring the (ii) the impact of an attack, and the awareness of (iii) its evolution by tracking the situation. According to [14], various factors are to be considered upon designing dashboards, being critical not to overload the dashboard with visual features, and to make prominent visual features to show important messages.

[15] defines the following method: (i) overview first, (ii) zoom and filter, then (iii) provide details-on-demand. First, it is necessary to grasp the situation and its possible outcomes to receive an overview, *e.g.*, analyzes a DDoS attack to determine whether it is necessary to request or accept/deny a cooperative defense. Furthermore, zoom and filter related to the detailed analysis of the specific event, *i.e.*, the filtering of unrelated events that can introduce unwanted noise into the analysis. Finally, further details (iii) can be provided on demand to the analyst.

Tools to visualize DDoS attacks vary from representations for non-technical audiences [16] to visualizations designed explicitly for cybersecurity analysts. While the former is related to the design of infographics and charts for a business audience, the latter aims at analyzing detailed attack data and transforming the data into a visual format optimized for cybersecurity analysts [17], [18]. Although there is not a one-size-fits-all visualization tool due to the subjective nature of the human perception process, research in visualization either leads to adapting existing visualization techniques to the cybersecurity or leading to novel ways to visualize cybersecurity data.

Furthermore, dashboards are commonly used to analyze data in (near) real-time. The focus is to understand the current state of a system concerning ongoing tasks or events of interest. [14] describes factors to consider during the design of dashboards, such as (i) overloading the dashboard with visual features, (b) to make prominent visual features to show essential messages, and (iii) design a dashboard within constraints, even though they still should be simple to view. Further, there should not be unnecessary decorations, or overused colors and other visual properties. Also, it is essential to note that event visualization is very much based on human perception, and as discussed in [15], it is important to keep the initial dashboard simple to provide details and metrics on demand.

## III. Dashboard

The goal of a collaborative defense is to be complementary with in-house defense mechanisms. Thus, the dashboard (*cf.* Figures 4 and 5 in the evaluation) is concerned with events and components related to information, such as the status of relevant services and individual attack reports submitted to the BC.

An AS can be in the role of a *M* or *T*, so both tabs are available to the network operator. These tabs display three columns based on a progressive state scheme (*i.e.*, it is possible to monitor the state of a mitigation service during its different stages) from left to right, in which the left lane is dedicated to new events [19]. The middle lane contains all events in progress and updates them accordingly to changes in their status. The right lane represents a log of all finished or declined elements. In more detail, the *REQUESTS_TAB* contains all incoming mitigation requests. The *ALARMS_TAB* contains all alarms triggered as soon as a pre-defined inbound traffic threshold is breached. In more detail, the *REQUESTS_TAB* contains all incoming mitigation requests.

Also, combining meaningful naming and appropriate coloring of states enhances the user experience for the security analyst. Therefore, it is crucial to standardize the use of colors in security visualization [20] to enhance rapid information processing. A classic example in this regard is the use of red for signaling events that require an action whereas yellow is typically used for signaling events that require attention. Colors were selected according to their meaning in the process context. For incoming mitigation requests, for example, state identifiers start with `MITIGATION_REQ_`$\star$, and for outgoing requests for mitigation start with `REQ_MITIGATION_`$\star$. In the following listing, $\star$ stands for both, incoming and outgoing mitigation requests.

1) States in red hues highlight highest importance and primary priority to the analyst, hence it is used to display a `NEW_ALARM` because they should be processed as soon as possible. Hence, if the target AS analyst decides to request help, ● `REQ_MITIGATION_REQUESTED` still needs to be monitored closely since the reaction of the mitigator AS has to be waited for.
2) States in yellow hues indicate a primary priority, but not as urgent as states in red, since the mitigator is not the target, but the origin of an attack. ● `NEW_MITIGATION_REQUEST` elements are displayed in the brightest yellow hue. Further, the states ● $\star$`_ACCEPTED` and ● $\star$`_IN_PROGRESS` are also yellow since they need monitoring and are not finished yet. Errors could still occur.
3) States in green and grey hues are secondary priority since they do not need any user interaction and were both already inspected by the analyst. Therefore, ● $\star$`_SUCCESSFUL` and ● $\star$`_DECLINED` do not draw immediate user attention with their coloring.

Figures 1 and 2 depict the state machines for both $T$'s and $M$'s perspective.

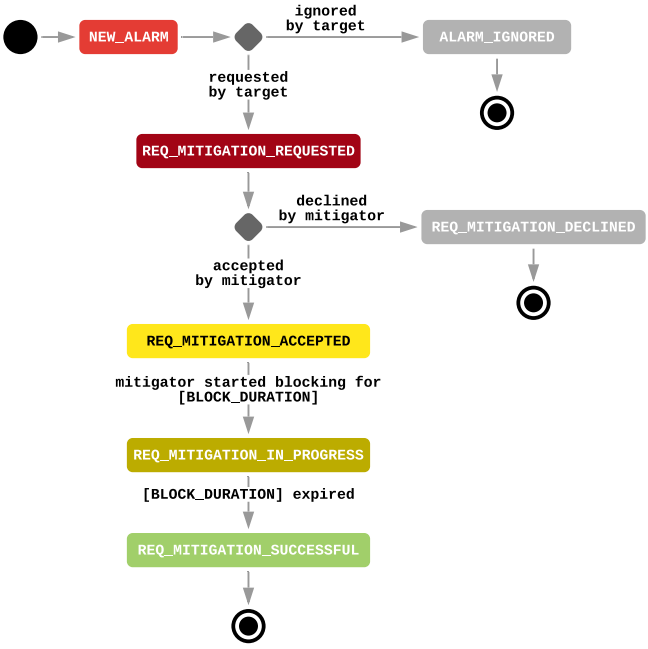



Fig. 1: State Diagram to Request Mitigation.

*1) Alarms Tab - Target Interaction Flow:* The start of the state diagram is triggered by a DDoS attack by nodes from other networks, operated by other ASes. In short, the AS of the attacked target will issue an attack report to the BC, which in turn will be retrieved by the AS of the attacking nodes, henceforth called Mitigator ($M$).

First, a network monitoring system sends traffic data to BloSS, which determines whether the pre-defined traffic thresholds are breached or not. For each breach, an attack report is sent to the BC and persisted with an initial status of ● NEW_ALARM if the same attack report hash has not been persisted yet.

The lifecycle of an attack report starts with the state ● NEW_ALARM as soon as a traffic breach occurs. The analyst of the target AS has to decide whether the attack should be reported or not (hence ignored). If the analyst ignores the alarm, the attack report changes state to ● ALARM_IGNORED and the lifecycle of the attack report ends. Otherwise, it is possible to request the cooperative mitigation and the state changes to ● REQUEST_MITIGATION_REQUESTED. This means that the attack report is submitted to the BC and retrieved by the mitigator ASes. It is important to note that at this point, an attack report with the state of ● NEW_MITIGATION_REQUEST is created on the mitigator's side (as soon as it was retrieved from the BC), hence starting the state machine in Figure 2.

Further, the mitigator decides whether or not to accept the request for mitigation. If the request is declined, the state changes to ● REQ_MITIGATION_DECLINED and the lifecycle of the attack report ends. Otherwise, the state changes to ● REQ_MITIGATION_ACCEPTED, meaning that the mitigator AS will block any harmful traffic. Now the states are congruent with the states of the mitigator (*cf.* Figure 2).

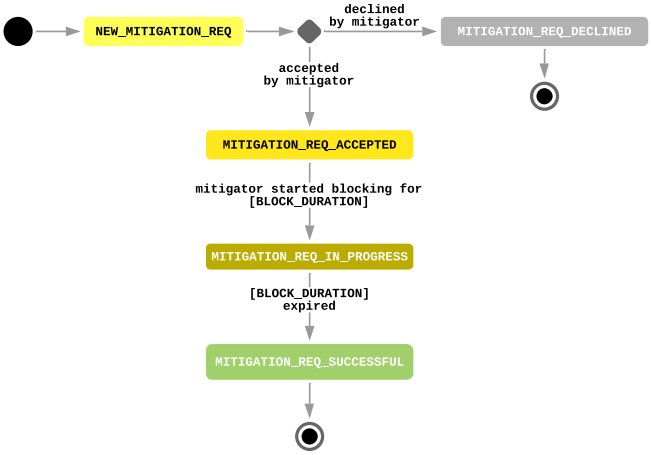



Fig. 2: State Diagram to Decide on Mitigation Requests.

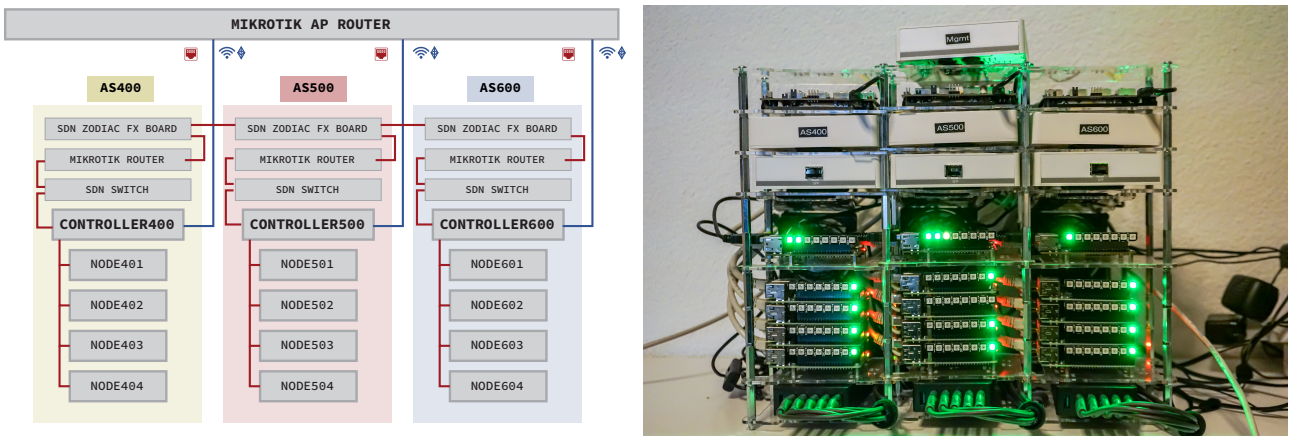
Therefore, from the moment that the mitigator starts blocking traffic, the state changes to ● REQ_MITIGATION_IN_PROGRESS. After the blocking time expires, the mitigation successfully ends, and the lifecycle of the attack report changes to the state of ● REQ_MITIGATION_SUCCESSFUL. Then, the AS of the attackers' origin will retrieve the attack reports and decide whether or not to accept the requests. From here on out, the process from the target's perspective is finished.

### A. Requests Tab - Mitigator Interaction Flow

An AS with the role of mitigator periodically checks for attacks signaled *i.e.*, reported on its SC. For example, when an AS is under attack, it can report the attackers directly on the SC of ASes that are responsible for the networks, from where the identified hosts originated. In this regard, BloSS instances are configured to periodically check for changes in the states of their Smart Contracts, *i.e.*, whether there are requests, and further creating events on the dashboard and updating whenever their status is changed. As soon as an attack report is reported and retrieved via IPFS, it is persisted in BloSS and displayed in the dashboard with an initial status of ● NEW_MITIGATION_REQUEST if the same attack report hash has not been persisted yet.

The analyst on the mitigator $M$ side has to decide whether the mitigation should be accepted. If the analyst declines, *e.g.*, provided incentives are not attractive, or the AS is unavailable to perform the requested mitigation; the attack report state is modified to ● MITIGATION_REQ_DECLINED and the lifecycle of the attack report ends. Otherwise, the state changes to an intermediary state of ● MITIGATION_REQ_ACCEPTED.

As soon as the AS performs the requested mitigation, traffic of reported attackers (hence mitigating the attack), the state changes to ● MITIGATION_REQ_IN_PROGRESS. When the mitigation is confirmed and relayed to the dashboard, the mitigation successfully ends and the lifecycle of the attack report changes to the state of ● MITIGATION_REQ_SUCCESSFUL.




(a) Schematic Representation

(b) BloSS Hardware

Fig. 3: BloSS schematic view and hardware prototype

## IV. Use Case Evaluation

The dashboard was deployed on a physical single board computer cluster (*cf.* Figure 3). Hardware and configuration details are presented in Subsecions IV-A and IV-B, and Use cases UC1 and UC2 on IV-D and IV-E, respectively.

### A. Hardware

Three isolated and identically configured ASes were built: AS 400, AS 500, and AS 600 with each AS consisting of four host nodes used to initiate the attack traffic, and two controllers, which host the BloSS as well as the Ethereum BC and IPFS [21] nodes. Hosts are based on Raspberry Pi Model B (RPi) and controllers use ASUS Tinker Board devices, which provide greater computational capacity than RPIs.

### B. Configuration

The schematic representation (*cf.* Figure 3) (a) shows the network setup in the BloSS hardware (b). The management network is configured via wireless between the three routers, the two auxiliary controllers, and the gateway. Furthermore, additional MikroTik routers and switches are necessary since the Zodiac FX switches only provide four ports, which is not sufficient to connect all hosts of an AS. The SDN controller responsible for the Zodiac FX switches is directly specified in the Zodiac FX Web interface.

### C. Evaluation Scenario

Hosts connected to AS 400 (4 hosts), AS 500 (4 hosts) and AS 600 (three hosts) will start a flood-based DDoS attack on a single host on AS 600. Controllers of all ASes are configured with a static inbound traffic threshold to determine whether there is an ongoing attack. Based on this, alarms on the AS 400 dashboard will show a warning about the attack and the operator can decide to request the cooperative mitigation or the ignore alarm (UC1). Then, if cooperative mitigation is requested, the dashboard on AS 600 will display the mitigation request (UC2).

### D. UC1 - Request Mitigation or Ignore Alarm

The precondition for UC1 is that all the BLoSS services are active and operating correctly (*cf.* Figure 4). In this regard, the left-hand side of the dashboard shows the interface (BloSS AS 400) that presents the status of services, which can have its services activated or deactivated modules based on a click. In the first step, as soon as the inbound traffic breaches the pre-defined threshold (*i.e.*, a DDoS attack is detected), alarms are sent to the dashboard, and the operator has to decide on whether to request or ignore these alarms. Then, the dashboard displays a message ● `NEW_ALARM`, which is seen in the Requests column in Figure 4, and the security analyst can decide on whether to request for collaborative defense or ignore the alarm.

In the following, the security analyst should decide whether to request the cooperative mitigation or ignore the alarm. If a mitigation is required, a request is sent to BloSS, which submits a transaction to the BC and the request is moved to the column ”In Progress” on the dashboard with status ● `REQUEST_MITIGATION_REQUESTED`. Then, the target operators on AS 500 and AS 600 can either decline the request for mitigation and the status of the attack_report changes to ● `REQ_MITIGATION_DECLINED` or accept the request for mitigation and the status of the attack_report changes to ● `REQ_MITIGATION_ACCEPTED`.

As soon as the involved mitigator involved starts blocking, *i.e.*, applying a mitigation action such as blackholing traffic or blocking hosts listed as attackers, the attack_report is also ● REQ_MITIGATION_IN_PROGRESS. After the expiration of the maximum block duration, the attack_report is completed, and thus, ends in the status ● `REQ_MITIGATION_SUCCESSFUL`. The history of requests, besides being registered in the BC (not disclosing the details, *e.g.*, blacklisted addresses), is available to all members of the alliance. The events involving each domain are also registered and grouped in the ”Log” column and can have their details visualized on demand.

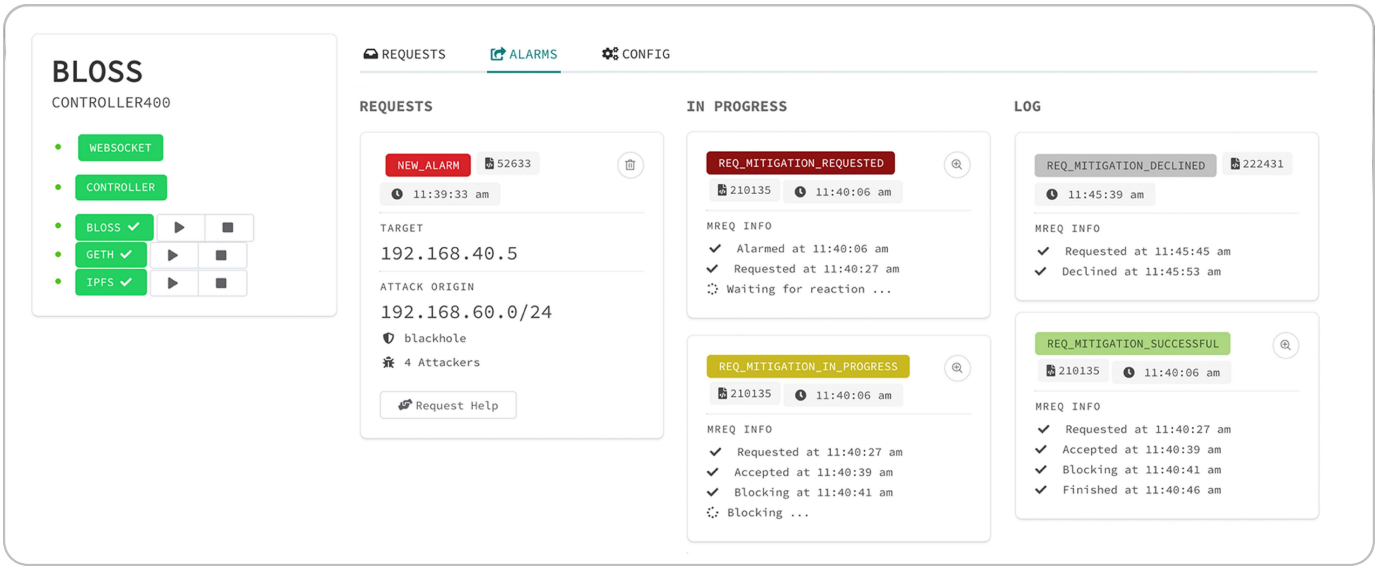



Fig. 4: `bloss-dashboard` UI running on AS 400 with all possible states of an example attack_report in the `ALARMS_TAB`

### *E. UC2 - Accept or Decline Mitigation Request*

This use case, in contrast to UC1, considers the mitigator's perspective available in the "Requests" tab (*cf.* Figure 5). Mitigation requests are periodically retrieved from the BC, which are relayed to `bloss-dashboard` to display to the operator with the status ● `NEW_MITIGATION_REQUEST`. Incoming requests can be grouped depending on the number of requests and the operator can click on specific events for more details.

At this point, an operator can decide to deny the request (and the attack_report's status changes to ● `MITIGATION_REQ_DECLINED`), or to accept the request (and the attack_report's status changes to ● `MITIGATION_REQ_ACCEPTED`). Once requests are accepted or denied, the dashboard forwards the request to BloSS, which submits the transaction to the BC. Then, the mitigation service has a deadline to be completed, which can be visualized in the time stamps registered on the dashboard. This service deadline is required for the mitigator to upload a proof of completion of the requested mitigation task, ensuring the effectiveness of a cooperative mitigation service.

Traffic from attacking hosts is mitigated, the attack_report's status changes to ● `MITIGATION_REQ_IN_PROGRESS`. At this point, the mitigator can act rationally and upload a proof or miss the upload. However, it is not possible to verify the truthfulness or the quality of the (proof of) service performed by $M$ (issue discussed in [22]). Even if the BC can preserve a transparent audit trail for all transactions, it cannot compensate for lack of ground-truth. This holds for the uploaded proof of service, as well as for user-defined, subjective ratings, in which there is no automated way to determine the truthfulness of proof or rating fully.

Once the service is completed, and the proof is uploaded (*e.g.*, log showing a mitigation action), or the mitigation deadline is expired, the service is considered complete, and the status is changed to ● `MITIGATION_REQ_SUCCESSFUL`. Similarly to UC1, all mitigation service events involving each domain are registered and grouped in the "Log" column, and its details can be visualized on demand. The transparency of actions recorded in the BC as well as their details locally on the requester and the mitigation on logs is useful in cases where a mitigation service does not have its effectiveness proven by the uploaded proof.

## V. Considerations and Future Work

Visualizing and classifying cybersecurity events is not a straightforward task. A cooperative defense adds a layer of complexity, in which not only should internal threats be analyzed and classified, but also threats of cooperative ASes. This paper presented a threat management dashboard providing a simple and objective interface for cybersecurity analysts. The dashboard allows for managing both incoming mitigation requests and requests for mitigation service, with the ability being possible to follow their progress in near real-time. Further, this paper complements the original architecture of BloSS, which is based on a permissioned BC with a Proof-of-Authority consensus, enabling the visualization and management of cooperative defense requests in a dashboard.

While command line interfaces provide greater agility and efficiency in the use of resources, a dashboard provides the operator an efficient situational analysis through its visual intuitiveness and the use of graphic elements. Hence, reaction speed is not an element as important as situational awareness, which favors the use of dashboards the context of security.

Future work includes improvements on the configuration tab to offer additional configuration possibilities *e.g.*, changing traffic-thresholds, the maximum number of alarms, mitigation requests to be visualized, and other parameters directly from

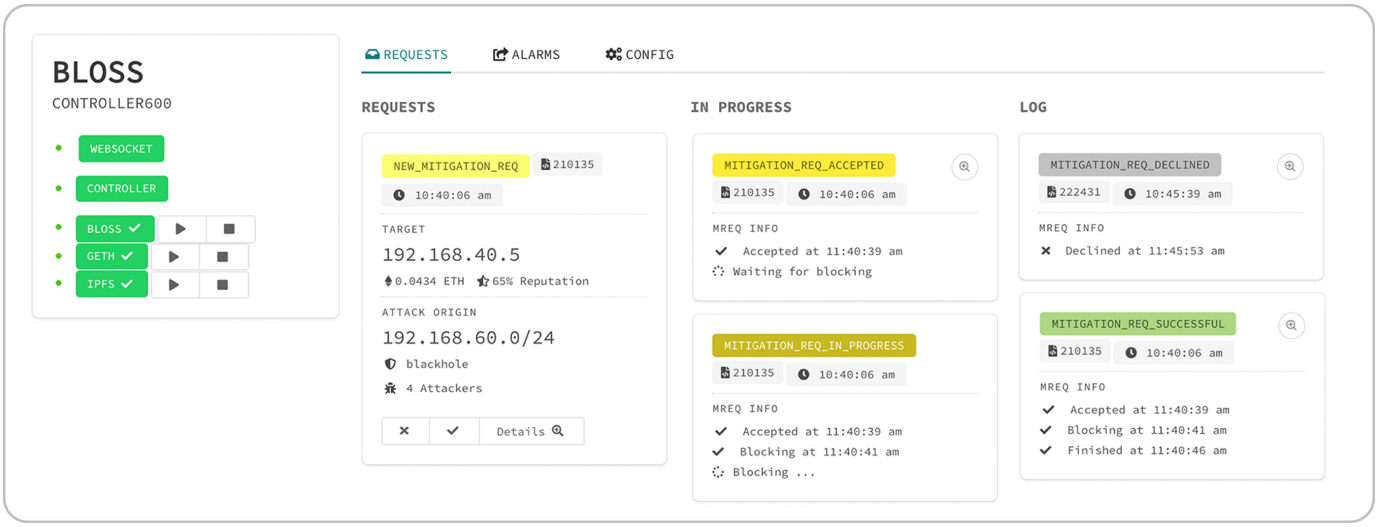



Fig. 5: `bloss-dashboard` UI running on AS 600 with all possible states of an example attack_report in the `REQUESTS_TAB`

the UI. Also, additional implementation should provide detailed insights into the reputation history of members involved in mitigation services.

## Acknowledgements

This paper was supported partially by (a) the University of Zurich UZH, Switzerland and (b) the European Union's Horizon 2020 Research and Innovation Program under grant agreement No. 830927, the Concordia project.

## References

[1] A. M. Odlyzko, "Internet traffic growth: Sources and implications," *Optical transmission systems and equipment for WDM networking II*, Vol. 5247. International Society for Optics and Photonics, 2003, pp. 1–16.

[2] A. Khalimonenko, O. Kupreev, and E. Badovskaya, "DDoS Attacks in Q1 2018," [Online] https://securelist.com/ddos-report-in-q1-2018/85373/, April 2018, last visit September 14, 2019.

[3] G. Oikonomou, J. Mirkovic, P. Reiher, and M. Robinson, "A Framework for a Collaborative DDoS Defense," *2006 22nd Annual Computer Security Applications Conference (ACSAC'06)*, December 2006, pp. 33–42.

[4] S. T. Zargar, J. Joshi, and D. Tipper, "A Survey of Defense Mechanisms Against Distributed Denial of Service (DDoS) Flooding Attacks," *IEEE Communications Surveys & Tutorials*, Vol. 15, No. 4, pp. 2046–2069, March 2013.

[5] T. Peng, C. Leckie, and K. Ramamohanarao, "Survey of Network-based Defense Mechanisms Countering the DoS and DDoS Problems," *ACM Computing Surveys (CSUR)*, Vol. 39, No. 1, p. 3, April 2007.

[6] B. Rodrigues, T. Bocek, and B. Stiller, "Enabling a Cooperative, Multi-domain DDoS Defense by a Blockchain Signaling System (BloSS)," *43rd IEEE Conference on Local Computer Networks (LCN 2018), Demonstration Track*, Singapore, Singapore, October 2017, pp. 1–3.

[7] B. Rodrigues, T. Bocek, A. Lareida, D. Hausheer, S. Rafati, and B. Stiller, "A Blockchain-based Architecture for Collaborative DDoS Mitigation with Smart Contracts," *IFIP International Conference on Autonomous Infrastructure, Management and Security*. Springer, Cham, 2017, pp. 16–29.

[8] D. Staheli, T. Yu, R. J. Crouser, S. Damodaran, K. Nam, D. O'Gwynn, S. McKenna, and L. Harrison, "Visualization Evaluation for Cyber Security: Trends and Future Directions," *11th Workshop on Visualization for Cyber Security (VizSec 2014)*, New York, NY, U.S.A., November 2014, pp. 49–56.

[9] X. Li, Q. Wang, L. Yang, and X. Luo, "Network Security Situation Awareness Method Based on Visualization," *Third International Conference on Multimedia Information Networking and Security (MINES 2011)*, Shanghai, China, November 2011, pp. 411–415.

[10] C. Killer, B. Rodrigues, and B. Stiller, "Security Management and Visualization in a Blockchain-based Collaborative Defense," *1st IEEE International Conference on Blockchain and Cryptocurrency (ICBC 19)*, pp. 1–4, 2019, Seoul, South Korea.

[11] P. Barford, M. Dacier, T. G. Dietterich, M. Fredrikson, J. Giffin, S. Jajodia, S. Jha, J. Li, P. Liu, P. Ning, X. Ou, D. Song, L. Strater, V. Swarup, G. Tadda, C. Wang, and J. Yen, *Cyber SA: Situational Awareness for Cyber Defense*. Boston, MA, U.S.A., Springer US, 2010, pp. 3–13.

[12] L. Hao, C. G. Healey, and S. E. Hutchinson, "Ensemble Visualization for Cyber Situation Awareness of Network Security Data," *2015 IEEE Symposium on Visualization for Cyber Security (VizSec)*, October, pp. 1–8.

[13] R. Marty, *Applied Security Visualization*, Addison-Wesley, Upper Saddle River, NJ, U.S.A., 2008.

[14] J. Jacobs and B. Rudis, *Data-Driven Security: Analysis, Visualization and Dashboards*, Wiley, Hoboken, NJ, U.S.A., 2014.

[15] B. Shneiderman, "A Grander Goal: A Thousand-fold Increase in Human Capabilities," *Educom review*, Vol. 32, No. 6, pp. 4 – 10, November 1997.

[16] AO Kaspersky Lab., "Cyberthreat Real-Time Map," [Online] https://cybermap.kaspersky.com, last visit September 14, 2019.

[17] S. McKenna, D. Staheli, C. Fulcher, and M. Meyer, "BubbleNet: A Cyber Security Dashboard for Visualizing Patterns," *Computer Graphics Forum (EuroVis 2016)*, Vol. 35, No. 3, Groningen, Netherlands, 2016, pp. 281–290.

[18] A. Yelizarov and D. Gamayunov, "Visualization of Complex Attacks and State of Attacked Network," *6th International Workshop on Visualization for Cyber Security (VizSec 2009)*, Atlantic City, NJ, USA, November 2009, pp. 1–9.

[19] N. Oza, F. Fagerholm, and J. Münch, "How does Kanban Impact Communication and Collaboration in Software Engineering Teams?" *6th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE 2013)*, San Francisco, CA, U.S.A., May 2013, pp. 125–128.

[20] J. Garae, R. K. L. Ko, and M. Apperley, "A Full-Scale Security Visualization Effectiveness Measurement and Presentation Approach," *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, August 2018, pp. 639–650.

[21] J. Benet, "IPFS - Content Addressed, Versioned, P2P File System," *CoRR*, Vol. abs/1407.3561, 2014.

[22] S. Mannhart, B. Rodrigues, E. Scheid, S. S. Kanhere, and B. Stiller, "Toward Mitigation-as-a-Service in Cooperative Network Defenses," *3rd IEEE Cyber Science and Technology Congress (CyberSciTech 2018)*, Athens, Greece, August 2018, pp. 362–367.